

# Python Dictionary

# Python Dictionary

- Python dictionary is an unordered collection of items.
- While other compound data types have only value as an element, a dictionary has a **key: value** pair.
- Dictionaries are optimized to retrieve values when the key is known.

# Creating a dictionary

- Creating a dictionary is as simple as placing items inside curly braces {} separated by comma.
- Each element in a dictionary is represented by a **key:value** pair.
- While values can be of any data type and can repeat, keys must be of immutable type and must be unique.

```
marks = {"Maths" : 45, "Science" : 23, "Social" : 38}
```



# Accessing elements from a dictionary

- While indexing is used with other container types to access values, dictionary uses keys. Key can be used either inside square brackets or with the `get()` method.
- The difference while using `get()` is that it returns **None** instead of **KeyError**, if the key is not found.

```
print(marks["Science"])
```

23

```
print(marks["Hindi"])
```

-----  
**KeyError**

```
<ipython-input-50-cadaa367708b> in  
----> 1 print(marks["Hindi"])
```

**KeyError: 'Hindi'**

```
print(marks.get("Maths"))
```

45

```
print(marks.get("Hindi"))
```

None



# How to change or add elements in a dictionary?

- Dictionary are mutable. We can add new items or change the value of existing items using assignment operator.
- If the key is already present, value gets updated, else a new key: value pair is added to the dictionary.

```
studentDetails = {"Name" : "Sikander" , "Course" : "PYTHON"}  
  
print(studentDetails)  
  
studentDetails["Course"] = "Data Science"  
  
print(studentDetails)  
  
studentDetails["Marks"] = 45  
  
print(studentDetails)
```



# Deleting elements from a dictionary

- We can remove a particular item in a dictionary by using the method pop(). This method removes an item with the provided key and returns the value.
- All the items can be removed at once using the clear() method.
- We can also use the del keyword to remove individual items or the entire dictionary itself.

```
print(studentDetails)
studentDetails.pop("Course")
print(studentDetails)
```

```
{'Name': 'Sikander', 'Course': 'Data Science', 'Marks': 45}
{'Name': 'Sikander', 'Marks': 45}
```

```
print(studentDetails)
studentDetails.clear()
print(studentDetails)
```

```
{'Name': 'Sikander', 'Course': 'Data Science', 'Marks': 45}
{}
```



# Deleting elements from a dictionary

```
squares = {1:1 , 2:4, 3:9, 4:16 , 5:25}  
print(squares)  
key = int(input("Enter key to delete"))  
del squares[key]  
print(squares)
```

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

```
Enter key to delete4
```

```
{1: 1, 2: 4, 3: 9, 5: 25}
```

# To delete random element

- The method, `popitem()` can be used to remove and return an arbitrary item (key, value) from the dictionary.
- Raises `KeyError` if the dictionary is empty.

```
squares = {1:1 , 2:4, 3:9, 4:16 , 5:25}  
print(squares)  
squares.popitem()  
print(squares)
```

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

```
{1: 1, 2: 4, 3: 9, 4: 16}
```



# Clear method and del statement

```
students = {1:"SIKANDER" , 2:"SATYA",3:"HUSSAIN"}
courses = {1:"EMBEDDED" , 2: "DATA SCIENCE" , 3:"WEB DEVELOPMENT"}

print('List of Students')
print(students)
print('List of Courses')
print(courses)

students.clear()

del courses

print('List of Students')
print(students)
print('List of Courses')
print(courses)
```

List of Students

{1: 'SIKANDER', 2: 'SATYA', 3: 'HUSSAIN'}

List of Courses

{1: 'EMBEDDED', 2: 'DATA SCIENCE', 3: 'WEB DEVELOPMENT'}

List of Students

{}

List of Courses

-----  
NameError

Traceback (most recent ca:



# Dictionary Membership Test

- We can test if a key is in a dictionary or not using the keyword `in`. Notice that membership test is for keys only, not for values.

```
squares = {1: 1, 3: 9, 5: 25, 7: 49, 9: 81}

print("1 in squares      :", 1 in squares)

print("2 not in squares :", 2 not in squares)

# membership tests for key only not value
print("49 in squares    :", 49 in squares)
```

```
1 in squares      : True
2 not in squares  : True
49 in squares     : False
```



# Iterating Through a Dictionary

- Using a for loop we can iterate through each key in a dictionary.

---

```
marks = {"Maths" : 45 , "Science" : 23 , "Social" : 38}
```

```
for subject in marks:  
    print(subject)
```

```
for subject in marks:  
    score = marks[subject]  
    print(subject, ":", score)
```

```
marks = {"Maths" : 45 , "Science" : 23 , "Social" : 38}

print("Keys : " , marks.keys())
print("Values : " , marks.values())

print("-----")
for subject, score in marks.items():
    print(subject, ":", score)
```



# Frequency Count using Dict

```
data = "PYTHON DICTIONARY IS AMAZING"
```

```
freq = {}
```

```
for c in data:
```

```
    freq[c] = freq.get(c , 0) + 1
```

```
for key in freq:
```

```
    print(key , freq[key])
```



# Python Dictionary Methods

Method	Description
<a href="#"><code>clear()</code></a>	Remove all items form the dictionary.
<a href="#"><code>copy()</code></a>	Return a shallow copy of the dictionary.
<a href="#"><code>fromkeys(seq[, v])</code></a>	Return a new dictionary with keys from <code>seq</code> and value equal to <code>v</code> (defaults to <code>None</code> ).
<a href="#"><code>get(key[,d])</code></a>	Return the value of <code>key</code> . If <code>key</code> doesnot exit, return <code>d</code> (defaults to <code>None</code> ).
<a href="#"><code>items()</code></a>	Return a new view of the dictionary's items (key, value).
<a href="#"><code>keys()</code></a>	Return a new view of the dictionary's keys.
<a href="#"><code>pop(key[,d])</code></a>	Remove the item with <code>key</code> and return its value or <code>d</code> if <code>key</code> is not found. If <code>d</code> is not provided and <code>key</code> is not found, raises <code>KeyError</code> .
<a href="#"><code>popitem()</code></a>	Remove and return an arbitrary item (key, value). Raises <code>KeyError</code> if the dictionary is empty.
<a href="#"><code>setdefault(key[,d])</code></a>	If <code>key</code> is in the dictionary, return its value. If not, insert <code>key</code> with a value of <code>d</code> and return <code>d</code> (defaults to <code>None</code> ).
<a href="#"><code>update([other])</code></a>	Update the dictionary with the key/value pairs from <code>other</code> , overwriting existing keys.
<a href="#"><code>values()</code></a>	Return a new view of the dictionary's values



# Built-in Functions with Dictionary

Function	Description
<a href="#"><code>len()</code></a>	Return the length (the number of items) in the dictionary.
<a href="#"><code>sorted()</code></a>	Return a new sorted list of keys in the dictionary.

```
squares = {1: 1, 3: 9, 5: 25, 7: 49, 9: 81}
```

```
# Output: 5  
print(len(squares))
```

```
# Output: [1, 3, 5, 7, 9]  
print(sorted(squares))
```

```
5  
[1, 3, 5, 7, 9]
```

---





# Motivation for Dictionaries

- # Definition of country and capital
- countries = ['spain', 'france', 'germany', 'norway', 'india']
- capitals = ['madrid', 'paris', 'berlin', 'oslo', 'delhi']
- I would like to print the capital of germany.
- # Get index of 'germany': ind\_ger
- # Use ind\_ger to print out capital of Germany

```
# Definition of country and capital
countries = ['spain', 'france', 'germany', 'norway', 'india']
capitals = ['madrid', 'paris', 'berlin', 'oslo', 'delhi']

ind_ger = countries.index('germany')

print(capitals[ind_ger])
```

# Create a Dictionary of Country and Capitals

```
countryCapitals = { 'spain': 'madrid',  
                    'france': 'paris',  
                    'germany': 'berlin',  
                    'norway': 'oslo'  
                  }  
  
print (countryCapitals)  
  
print (countryCapitals[ 'germany' ])
```



# Dictionary Manipulation

- If you know how to access a dictionary, you can also assign a new value to it. To add a new key-value pair to countryCapitals, you can use something like this:

countryCapitals['india'] = 'delhi'

```
countryCapitals = {'spain': 'madrid',  
                  'france': 'paris',  
                  'germany': 'berlin',  
                  'norway': 'oslo'  
                  }  
print(countryCapitals)  
  
#Add India to dictionary  
countryCapitals['india'] = 'delhi'  
  
print(countryCapitals)
```

# Dictionary Manipulation

- To remove a entry from dictionary, we can use del method.
- Remove **norway**
- `del (countryCapitals['norway'])`

---

```
countryCapitals = {'spain': 'madrid',  
                  'france': 'paris',  
                  'germany': 'berlin',  
                  'norway': 'oslo',  
                  'inida': 'delhi'  
                  }
```

```
print(countryCapitals)
```

```
#Remove Norway from dictionary
```

```
del (countryCapitals['norway'])
```

```
print(countryCapitals)
```



# Search for Keys in Dictionary

```
countryCapitals = {'spain': 'madrid',  
                  'france': 'paris',  
                  'germany': 'berlin',  
                  'norway': 'oslo',  
                  'india': 'delhi'  
                  }  
  
print(countryCapitals)  
  
#To search if a particular country is there in dictionary  
key = input("Enter the country to search : ")  
if (key in countryCapitals) :  
    print('Entry present - Capital = ' , countryCapitals[key])  
else:  
    print('The specified country not present')
```

# Multiple values for each key

- Create a Dictionary which records the major cities of each state.
- {state : [major cities] }

```
majorcities = {'karnataka' : ['bengaluru' , 'mysore' , 'hubli' ] ,  
               'tamilnadu' : ['chennai' , 'coimbatore' , 'madurai' ] ,  
               'maharashtra' : ['mumbai' , 'pune' ]  
               }
```

```
print(majorcities)
```

```
print('\nCities of Tamilnadu')  
print(majorcities['tamilnadu'])
```

```
print('\nDifferent states ' )  
print(majorcities.keys())
```



# Dictionary of dictionaries

- Dictionaries can contain key:value pairs where the values are again dictionaries.

```
india = { 'karnataka': { 'capital':'bengaluru', 'population':46.77 },  
          'tamilnadu': { 'capital':'chennai', 'population':66.03 },  
          'maharashtra':{ 'capital':'mumbai', 'population':80.62 },  
          'kerala':     { 'capital':'thiruvananthapuram', 'population':5.084 }  
        }
```

```
# Print out the capital of kerala
```

```
print('Capital of kerala is ', india['kerala']['capital'])
```

```
# Create sub-dictionary data
```

```
data = { 'capital':'hyderabad' , 'population':59.83 }
```

```
# Add data to india under key 'telangana'
```

```
india['telangana'] = data
```

```
# Print india
```

```
print('\n', india)
```

# List vs Dictionary

List	Dictionary
Select, update and remove: [ ]	Select, update and remove: [ ]
Indexed by range of numbers	Indexed by unique keys
Collection of values order matters select entire subsets	Lookup table with unique keys